

Data Science: Churn Analysis

Detect customer churn based on use of dating platform

Marilson Campos

Version: 1.01 - January, 2020

Contents

1	Introduction	2
2	Environment Setup	3
2.1	Loading the R packages used in project.	3
3	Helper functions.	3
4	Preparing the datasets.	4
5	Basic Exploratory Data Analysis (EDA)	5
5.1	Looking at highly correlated variables	6
5.1.1	Selecting the variables with large correlation coefficients.	6
6	Dropping variables	6
7	Build a Logistic Regression Model	7
7.1	Model review	8
7.1.1	Significant variables at 0.05 level.	8
7.2	Nonsignificant variables at 0.05 level	8
8	Model including only the significant variables	9
8.1	preparing a dataset containing only the data we need.	9
8.2	Build the model using the new data.	9
9	Predicting churn	10
10	Possible improvements	10

1 Introduction

The objective of this project try to predict cancelations of an online dating site based on the behaviour of the customers on the platform. The variables used in the models represent real types of events and occur between two users.

For each event there are variables marked with the prefix ‘i_’ meaning the current user received an event or with ‘o_’ prefix fix the user initiated (outgoing). For example a user can ‘send a chat request’ to another user. This action will create two records one for the user that sent the event with a prefix ‘o_’ and onother one to the user receiving the request with the prefix ‘i_’.

There is also a measurement of engagement between users. This is marked with strings of type dd, where and are numbers.

For example, chat request marked as ‘d2d1’ means that the user that send that request is interacting for the second time and the the other user already gave him/her a response. As these numbers are higher on events it means that users are engaging in conversations.

Our goal is to try to learn these signals and identify users that are likely to cancel the service.

We will be using a variable called cancel_07 that marks users that canceled withing seven days of subscription.

We are following the sequence of steps as described below:

1. Project setup and dataset load
2. Definition of few helper funcions
3. Exploratory Data Analysis
4. Indentification of highly correlated variables.
5. Build Initial logistic regression
6. Validate the significance of varialbes
7. Build a model using the significant variables
8. Present results

2 Environment Setup

2.1 Loading the R packages used in project.

```
repo_url <- "http://cran.us.r-project.org"
if(!require(kableExtra))
  install.packages("kableExtra", repos = repo_url)
if(!require(tidyverse))
  install.packages("tidyverse", repos = repo_url)
if(!require(data.table))
  install.packages("data.table", repos = repo_url)
if(!require(caret))
  install.packages("caret", repos = repo_url)
```

3 Helper functions.

```
cor_prob <- function (X, dfr = nrow(X) - 2) {
  R <- cor(X, use="pairwise.complete.obs")
  above <- row(R) < col(R)
  r2 <- R[above]^2
  Fstat <- r2 * dfr/(1 - r2)
  R[above] <- 1 - pf(Fstat, 1, dfr)
  R[row(R) == col(R)] <- NA
  R
}

flatten_square_matrix <- function(m) {
  if( (class(m) != "matrix") | (nrow(m) != ncol(m))) stop("Must be a square matrix.")
  if(!identical(rownames(m), colnames(m))) stop("Row and column names must be equal.")
  ut <- upper.tri(m)
  data.frame(i = rownames(m)[row(m)[ut]], j = rownames(m)[col(m)[ut]],
             cor=t(m)[ut], p=m[ut])
}

order_by_pvalue <- function(fit.data, selected) {
  d1 <- data.frame(summary(fit.data)$coeff)
  dta <- cbind(variable=rownames(d1), d1)
  dta <- dta[selected, ]
  names(dta) <- c('Variable', 'Estimate', 'Std.Error', 'z.value', 'pvalue')
  selected.ds <- dta %>%
    filter(Variable != '(Intercept') %>%
    arrange(pvalue)
  names(selected.ds)[5] <- 'p-value'
  return(selected.ds)
}

separate.variables <- function(fit.data, alpha.level) {
  signif.idx <- summary(fit.data)$coeff[,4] < alpha.level
  signif.idx[0] <- TRUE
  result <- list(significant = order_by_pvalue(fit.data, signif.idx),
                 nonsignificant = order_by_pvalue(fit.data, !signif.idx))
  return(result)
}
```

4 Preparing the datasets.

Here we are loading the train dataset. Since the dataset is from a real system, we are ignoring irrelevant columns and using the field relevant to the churn analysis considering cancelation within 7 days.

```
load_clean_data <- function (ds_filename) {  
  temp <- read.table(ds_filename, sep=",", header=TRUE)  
  temp['i_pmf_7d_after_sub'] <- temp['i_pmfcan_7d_after_sub'] + temp['i_pmfcus_7d_after_sub']  
  temp_tb <- as_tibble(temp) %>%  
    filter(duration == 1 & gender == 1 & iswinback==0) %>%  
    select(  
      cancel_07,  
      i_fr_7d_after_sub, i_ch_7d_after_sub, i_fl_7d_after_sub,  
      i_pv_7d_after_sub, i_wr_7d_after_sub, i_fl_from_blocked_7d_after_sub,  
      i_wi_from_blocked_7d_after_sub,  
      o_wi_7d_after_sub, o_fr_7d_after_sub, o_ch_7d_after_sub, o_fl_7d_after_sub,  
      o_pv_7d_after_sub,  
      o_ciy_7d_after_sub, o_caru_7d_after_sub, o_za_zm_7d_after_sub,  
      o_zd_7d_after_sub, o_fl_blocked_7d_after_sub,  
      o_wi_blocked_7d_after_sub, photo_ver_7d_after_sub,  
      d2d2_blocked_7d_after_sub,  
      i_fl_pm_7d_after_sub, i_gmf_7d_after_sub, i_pmf_7d_after_sub,  
      o_gmf_7d_after_sub, o_pmfcan_7d_after_sub, o_pmfcus_7d_after_sub,  
      o_cin_7d_after_sub,  
      invi_7d_after_sub)  
  # model1_ds <- na.omit(model1_ds)  
  temp_tb  
}  
  
# Loading the data -----  
work_dir <- '/tmp/churn'  
train_filename <- paste(work_dir, '/', 'train_data.csv', sep = '')  
test_filename <- paste(work_dir, '/', 'test_data.csv', sep = '')  
train_tb <- load_clean_data(train_filename)  
test_tb <- load_clean_data(test_filename)
```

5 Basic Exploratory Data Analysis (EDA)

Lets take a look into our train dataset.

```
str(train_tb)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame': 35372 obs. of 29 variables:  
## $ cancel_07 : int 0 0 1 0 1 0 0 1 0 0 ...  
## $ i_fr_7d_after_sub : int 7 2 10 8 0 20 20 0 10 11 ...  
## $ i_ch_7d_after_sub : int 12 53 17 4 0 12 31 0 154 68 ...  
## $ i_fl_7d_after_sub : int 26 52 26 15 0 51 72 0 31 62 ...  
## $ i_pv_7d_after_sub : int 74 215 242 198 0 459 563 0 397 469 ...  
## $ i_wr_7d_after_sub : int 0 0 1 0 0 0 1 0 0 0 ...  
## $ i_fl_from_blocked_7d_after_sub: int 0 10 3 0 0 5 1 0 3 2 ...  
## $ i_wi_from_blocked_7d_after_sub: int 0 1 0 0 0 2 1 0 2 0 ...  
## $ o_wi_7d_after_sub : int 0 1 10 0 0 0 19 0 0 7 ...  
## $ o_fr_7d_after_sub : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ o_ch_7d_after_sub : int 10 27 7 3 0 4 8 0 223 106 ...  
## $ o_fl_7d_after_sub : int 11 61 30 1 0 48 61 0 26 89 ...  
## $ o_pv_7d_after_sub : int 47 181 48 25 0 134 493 0 36 372 ...  
## $ o_ciy_7d_after_sub : int 0 1577 81 0 0 1 4 0 0 41 ...  
## $ o_caru_7d_after_sub : int 0 0 5 0 0 0 0 0 0 0 ...  
## $ o_za_zm_7d_after_sub : int 7 67 0 0 0 3 1 0 7 26 ...  
## $ o_zd_7d_after_sub : int 79 1 1 1 0 8 6 0 43 26 ...  
## $ o_fl_blocked_7d_after_sub : int 0 4 3 0 0 4 3 0 0 0 ...  
## $ o_wi_blocked_7d_after_sub : int 0 0 2 0 0 0 0 0 0 0 ...  
## $ photo_ver_7d_after_sub : int 0 0 0 0 0 1 0 0 0 0 ...  
## $ d2d2_blocked_7d_after_sub : int 0 1 6 0 0 6 4 0 0 0 ...  
## $ i_fl_pm_7d_after_sub : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ i_gmf_7d_after_sub : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ i_pmf_7d_after_sub : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ o_gmf_7d_after_sub : int 0 0 0 0 0 0 0 0 0 181 ...  
## $ o_pmfcan_7d_after_sub : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ o_pmfccus_7d_after_sub : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ o_cin_7d_after_sub : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ invi_7d_after_sub : int 0 0 0 0 0 0 0 0 0 0 ...
```

As we can see our dataset has lots of variables representing the count of events between two users. The prefix ‘o_’ means that the user initiated the action while the prefix ‘i_’ means that the user received an event.

Also, we can see pairs of dd strings in the variable names. These represents that how deep the connection between two users. For example a d2d1 means that the user initiated 2 events and received 1 event from the other user.

5.1 Looking at highly correlated variables

5.1.1 Selecting the variables with large correlation coefficients.

Lets look at pairs of variables that have a correlation coefficients higher than 0.80.

```
cor_results <- flatten_square_matrix(cor_prob(train_tb))
cor_table <- cor_results %>%
  select(i,j,cor) %>%
  filter(abs(cor) > 0.80) %>%
  arrange(i,desc(cor))
names(cor_table) <- c('Variable 1', 'Variable 2', 'Correlation Coef.')
kable(cor_table) %>%
  kable_styling(position = "center", full_width = F)
```

Variable 1	Variable 2	Correlation Coef.
i_ch_7d_after_sub	o_ch_7d_after_sub	0.9108487
i_fr_7d_after_sub	i_pv_7d_after_sub	0.8163197

As we can see there are many variables pairs that are highly correlated.

Based on the list above we are selecting one of the variables of each pair to keep and will drop the other variable.

6 Dropping variables

Since the correlation coefficient of the 'o_ch_7d_after_sub' variable, we've are dropping from the model:

```
train_tb_v1 <- train_tb %>% select (-c(o_ch_7d_after_sub))
```

7 Build a Logistic Regression Model

```
logit_model_v1 <- glm(formula = cancel_07 ~ ., family = binomial, data = train_tb_v1)
summary(logit_model_v1)

##
## Call:
## glm(formula = cancel_07 ~ ., family = binomial, data = train_tb_v1)
##
## Deviance Residuals:
##      Min      1Q  Median      3Q     Max 
## -2.1174 -0.7735 -0.4761 -0.0664  5.1453 
##
## Coefficients: (1 not defined because of singularities)
##                               Estimate Std. Error z value Pr(>|z|)    
## (Intercept)                 0.0351844  0.0209824  1.677 0.093571 .
## i_fr_7d_after_sub           0.0010850  0.0019438  0.558 0.576735  
## i_ch_7d_after_sub           0.0018347  0.0002934  6.253 4.02e-10 ***
## i_fl_7d_after_sub           0.0009006  0.0005248  1.716 0.086140 .  
## i_pv_7d_after_sub          -0.0019715  0.0001427 -13.811 < 2e-16 ***
## i_wr_7d_after_sub          0.0003826  0.0122257  0.031 0.975032  
## i_fl_from_blocked_7d_after_sub 0.0039660  0.0039630  1.001 0.316952  
## i_wi_from_blocked_7d_after_sub -0.0463828  0.0216592 -2.141 0.032235 *  
## o_wi_7d_after_sub          -0.0047340  0.0023629 -2.003 0.045129 *  
## o_fr_7d_after_sub           0.0029846  0.0022771  1.311 0.189950  
## o_fl_7d_after_sub           -0.0070699  0.0007258 -9.741 < 2e-16 *** 
## o_pv_7d_after_sub           -0.0032435  0.0001874 -17.312 < 2e-16 *** 
## o_ciy_7d_after_sub          -0.0042553  0.0006317 -6.736 1.62e-11 *** 
## o_caru_7d_after_sub         -0.0308006  0.0184955 -1.665 0.095853 .  
## o_za_zm_7d_after_sub        -0.0339642  0.0029801 -11.397 < 2e-16 *** 
## o_zd_7d_after_sub           -0.0022746  0.0011339 -2.006 0.044864 *  
## o_fl_blocked_7d_after_sub   -0.0023231  0.0056159 -0.414 0.679119  
## o_wi_blocked_7d_after_sub   0.0863243  0.0588549  1.467 0.142450  
## photo_ver_7d_after_sub     -0.1665046  0.0273146 -6.096 1.09e-09 *** 
## d2d2_blocked_7d_after_sub   0.0005192  0.0011130  0.466 0.640868  
## i_fl_pm_7d_after_sub        0.0193759  0.0057212  3.387 0.000707 *** 
## i_gmf_7d_after_sub          2.4882529  0.2663090  9.343 < 2e-16 *** 
## i_pmf_7d_after_sub          1.2196640  1.2150030  1.004 0.315458  
## o_gmf_7d_after_sub          -0.0002524  0.0001314 -1.921 0.054723 .  
## o_pmfcan_7d_after_sub       -0.0011225  0.0039038 -0.288 0.773700  
## o_pmfcus_7d_after_sub       -0.0189877  0.0188061 -1.010 0.312660  
## o_cin_7d_after_sub           NA          NA          NA          NA      
## invi_7d_after_sub           0.0067465  0.0366145  0.184 0.853811 
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 35595  on 32207  degrees of freedom
## Residual deviance: 29968  on 32181  degrees of freedom
## (3164 observations deleted due to missingness)
## AIC: 30022
##
## Number of Fisher Scoring iterations: 7
```

7.1 Model review

7.1.1 Significant variables at 0.05 level.

```
model_v1_vars <- separate.variables(logit_model_v1, 0.05)
kable(model_v1_vars$significant) %>%
  kable_styling(position = "center", full_width = F)
```

Variable	Estimate	Std.Error	z.value	p-value
o_pv_7d_after_sub	-0.0032435	0.0001874	-17.311728	0.0000000
i_pv_7d_after_sub	-0.0019715	0.0001427	-13.810743	0.0000000
o_za_zm_7d_after_sub	-0.0339642	0.0029801	-11.396977	0.0000000
o_fl_7d_after_sub	-0.0070699	0.0007258	-9.740728	0.0000000
i_gmf_7d_after_sub	2.4882529	0.2663090	9.343480	0.0000000
o_ciy_7d_after_sub	-0.0042553	0.0006317	-6.736471	0.0000000
i_ch_7d_after_sub	0.0018347	0.0002934	6.253130	0.0000000
photo_ver_7d_after_sub	-0.1665046	0.0273146	-6.095821	0.0000000
i_fl_pm_7d_after_sub	0.0193759	0.0057212	3.386677	0.0007074
i_wi_from_blocked_7d_after_sub	-0.0463828	0.0216592	-2.141482	0.0322352
o_zd_7d_after_sub	-0.0022746	0.0011339	-2.005923	0.0448645
o_wi_7d_after_sub	-0.0047340	0.0023629	-2.003448	0.0451293

7.2 Nonsignificant variables at 0.05 level

```
kable(model_v1_vars$nonsignificant) %>%
  kable_styling(position = "center", full_width = F)
```

Variable	Estimate	Std.Error	z.value	p-value
o_gmf_7d_after_sub	-0.0002524	0.0001314	-1.9210680	0.0547231
i_fl_7d_after_sub	0.0009006	0.0005248	1.7161211	0.0861399
o_caru_7d_after_sub	-0.0308006	0.0184955	-1.6653010	0.0958527
o_wi_blocked_7d_after_sub	0.0863243	0.0588549	1.4667295	0.1424497
o_fr_7d_after_sub	0.0029846	0.0022771	1.3107283	0.1899496
o_pmfccus_7d_after_sub	-0.0189877	0.0188061	-1.0096555	0.3126604
i_pmf_7d_after_sub	1.2196640	1.2150030	1.0038362	0.3154576
i_fl_from_blocked_7d_after_sub	0.0039660	0.0039630	1.0007406	0.3169522
i_fr_7d_after_sub	0.0010850	0.0019438	0.5581597	0.5767353
d2d2_blocked_7d_after_sub	0.0005192	0.0011130	0.4664858	0.6408678
o_fl_blocked_7d_after_sub	-0.0023231	0.0056159	-0.4136663	0.6791185
o_pmfccan_7d_after_sub	-0.0011225	0.0039038	-0.2875390	0.7736996
invi_7d_after_sub	0.0067465	0.0366145	0.1842580	0.8538111
i_wr_7d_after_sub	0.0003826	0.0122257	0.0312979	0.9750320

As we can see there are many variables that are clearly not significant at p=0.05.

So our next step is to build a model using only the significant variables next.

8 Model including only the significant variables

8.1 preparing a dataset containing only the data we need.

```
extract_ds_v2 <- function(ds) {  
  ds %>%  
    select(cancel_07,  
          o_pv_7d_after_sub, i_pv_7d_after_sub, o_za_zm_7d_after_sub,  
          o_fl_7d_after_sub, i_gmf_7d_after_sub, o_ciy_7d_after_sub,  
          i_ch_7d_after_sub, photo_ver_7d_after_sub, i_fl_pm_7d_after_sub,  
          i_wi_from_blocked_7d_after_sub, o_zd_7d_after_sub, o_wi_7d_after_sub)  
}  
model_v1_vars$significant$Variable  
  
## [1] o_pv_7d_after_sub           i_pv_7d_after_sub  
## [3] o_za_zm_7d_after_sub       o_fl_7d_after_sub  
## [5] i_gmf_7d_after_sub         o_ciy_7d_after_sub  
## [7] i_ch_7d_after_sub          photo_ver_7d_after_sub  
## [9] i_fl_pm_7d_after_sub        i_wi_from_blocked_7d_after_sub  
## [11] o_zd_7d_after_sub          o_wi_7d_after_sub  
## 27 Levels: (Intercept) d2d2_blocked_7d_after_sub ... photo_ver_7d_after_sub  
train_tb_v2 <- extract_ds_v2(train_tb_v1)  
test_tb_v2 <- extract_ds_v2(test_tb)
```

8.2 Build the model using the new data.

```
logit_model_v2 <- glm(formula = cancel_07 ~ ., family = binomial, data = train_tb_v2)  
summary(logit_model_v2)  
  
##  
## Call:  
## glm(formula = cancel_07 ~ ., family = binomial, data = train_tb_v2)  
##  
## Deviance Residuals:  
##      Min        1Q    Median        3Q       Max  
## -2.1326   -0.8020   -0.4676    1.1343    5.3119  
##  
## Coefficients:  
##                               Estimate Std. Error z value Pr(>|z|)  
## (Intercept)            1.023e-01  1.892e-02  5.407  6.41e-08 ***  
## o_pv_7d_after_sub     -3.529e-03  1.811e-04 -19.484  < 2e-16 ***  
## i_pv_7d_after_sub     -1.844e-03  9.178e-05 -20.091  < 2e-16 ***  
## o_za_zm_7d_after_sub  -3.478e-02  2.936e-03 -11.844  < 2e-16 ***  
## o_fl_7d_after_sub     -6.277e-03  6.122e-04 -10.253  < 2e-16 ***  
## i_gmf_7d_after_sub    2.250e+00  2.335e-01   9.637  < 2e-16 ***  
## o_ciy_7d_after_sub    -4.728e-03  6.294e-04  -7.511  5.85e-14 ***  
## i_ch_7d_after_sub     1.956e-03  2.858e-04   6.843  7.73e-12 ***  
## photo_ver_7d_after_sub -1.900e-01  2.688e-02  -7.069  1.56e-12 ***  
## i_fl_pm_7d_after_sub   1.858e-02  5.640e-03   3.295  0.000985 ***  
## i_wi_from_blocked_7d_after_sub -3.812e-02  2.097e-02  -1.818  0.069071 .  
## o_zd_7d_after_sub     -2.592e-03  1.112e-03  -2.332  0.019723 *  
## o_wi_7d_after_sub     -5.155e-03  1.724e-03  -2.990  0.002790 **  
## ---
```

```

## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 40590  on 35222  degrees of freedom
## Residual deviance: 33818  on 35210  degrees of freedom
## (149 observations deleted due to missingness)
## AIC: 33844
##
## Number of Fisher Scoring iterations: 7

```

As we can see now this model is simpler and easier to understand and optimize.

9 Predicting churn

```

test_input <- test_tb_v2
test_input$pred_prob <- predict(logit_model_v2, test_tb_v2, type = "response")

test_input <- test_input %>%
  mutate(model_pred = 1 * (pred_prob > 0.53) + 0)
test_input$model_pred[is.na(test_input$model_pred)] <- 0
test_input <- test_input %>% mutate(accurate = 1 * (model_pred == cancel_07) + 0)
model_accuracy_pct <- sum(test_input$accurate)/nrow(test_input) * 100

accuracy_df <- data.frame(model = "Simple Model (Avg Rating)", rmse = round(model_accuracy_pct,digits=3))
names(accuracy_df) <- c("Metric", "Value")
kable(accuracy_df) %>%
  kable_styling(position = "center", full_width = F)

```

Metric	Value
Simple Model (Avg Rating)	74.3

As we can see our model in predicting churn correctly about 75% of time. This is good enough to use this model to select the users more likely to churm and give them some addition support and/or offer them some kind of dicount.

10 Possible improvements

- a. We could look at some of the variables that dont have a linear relationship with the probability of churn and transform them using a function like $\log(x)+1$.
- b. We could also expand the model include combination of events.